

Beyond L2-Loss Functions for Learning Sparse Models

Karthikeyan Natesan
Ramamurthy
IBM Thomas J. Watson
Research Center
1101 Kitchawan Road
Yorktown Heights, NY, USA
knatesa@us.ibm.com

Aleksandr Y. Aravkin
IBM Thomas J. Watson
Research Center
1101 Kitchawan Road
Yorktown Heights, NY, USA
saravkin@us.ibm.com

Jayaraman J.
Thiagarajan
Lawrence Livermore National
Laboratory
7000 East Ave
Livermore, CA, USA
jayaramanthi1@llnl.gov

ABSTRACT

Incorporating sparsity priors in learning tasks can give rise to simple, and interpretable models for complex high dimensional data. Sparse models have found widespread use in structure discovery, recovering data from corruptions, and a variety of large scale unsupervised and supervised learning problems. Assuming the availability of sufficient data, these methods infer dictionaries for sparse representations by optimizing for high-fidelity reconstruction. In most scenarios, the reconstruction quality is measured using the squared Euclidean distance, and efficient algorithms have been developed for both batch and online learning cases. However, new application domains motivate looking beyond conventional loss functions. For example, robust loss functions such as ℓ_1 and Huber are useful in learning outlier-resilient models, and the quantile loss is beneficial in discovering structures that are the representative of a particular quantile. These new applications motivate our work in generalizing sparse learning to a broad class of convex loss functions. In particular, we consider the class of piecewise linear quadratic (PLQ) cost functions that includes Huber, as well as ℓ_1 , quantile, Vapnik, hinge loss, and smoothed variants of these penalties. We propose an algorithm to learn dictionaries and obtain sparse codes when the data reconstruction fidelity is measured using any smooth PLQ cost function. We provide convergence guarantees for the proposed algorithm, and demonstrate the convergence behavior using empirical experiments. Furthermore, we present three case studies that require the use of PLQ cost functions: (i) robust image modeling, (ii) tag refinement for image annotation and retrieval and (iii) computing empirical confidence limits for subspace clustering.

Keywords

dictionary learning, sparse representation, robust penalties, piecewise linear quadratic, convex optimization

1. INTRODUCTION

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

KDD 2014 New York City, NY USA

Copyright 20XX ACM X-XXXXX-XX-X/XX/XX ...\$15.00.

Deriving predictive inference from data requires both approximating the generating process using a model, and estimating model parameters from input data and the observed responses. The generating process can be approximated as $y_i \approx f(x_i; a)$ where $x_i \in \mathbb{R}^K$ is the input data sample, $y_i \in \mathbb{R}$ is the corresponding response, f is the assumed model and $a \in \mathbb{R}^K$ are the parameters. When f is linear, this representation reduces to the classical linear model $y_i \approx x_i^T a$. Given the vector of observed responses $y \in \mathbb{R}^T$, and input data matrix $X = [x_1 x_2 \dots x_T] \in \mathbb{R}^{K \times T}$, the parameters a can be estimated using linear regression, where the ℓ_2 loss on the residual $r = y - X^T a$ is minimized. The complexity of the linear model can be reduced by shrinking the small entries in a to zero [21]. This approach gives a sparse linear model, where only a small fraction of the parameters are non-zero (and hence active). Sparse parameters allow improved model interpretability because of parsimony []. From the viewpoint of statistical learning theory, sparsity also improves the generalizability, and hence usefulness of the model.

When the observations are high-dimensional, we can denote each observation vector as $y \in \mathbb{R}^M$, and assume that it can be approximated using a sparse linear combination of representative columns in the *dictionary* matrix $D \in \mathbb{R}^{M \times K}$. The parameter vector a , also referred to as the *sparse code* of y , can be obtained by solving an optimization problem of the form

$$\min_a \rho_1(y - Da) + \lambda \rho_2(a) \quad (1)$$

where ρ_1 is the *loss function* that measures the distance between y and Da , ρ_2 is the sparsity *regularizer* on a , and λ is the regularization penalty that controls the trade-off between loss and regularization. The choice of loss function ρ_1 corresponds to the noise or deviation model for the discrepancy between the observed and predicted data. Sparse models have had widespread applications in speech and audio processing [19, 39, 18], image analysis and recovery [42, 14, 40], compressive sampling [12], blind source separation [50, 28, 20], unsupervised [35, 10], supervised [22, 23, 24, 30], semi-supervised [47] and transfer learning [33, 31].

So far, we have assumed that a pre-defined dictionary D is available for sparse coding. However, given a set of T observations, $\{y_i\}_{i=1}^T$, where T is sufficiently large, the dictionary D can be adapted from the data itself, by jointly minimizing the sum of T objectives in (1) over D and $\{a_i\}_{i=1}^T$. Additional constraints may also be placed on the dictionary and the sparse codes. Most of the existing dictionary learning

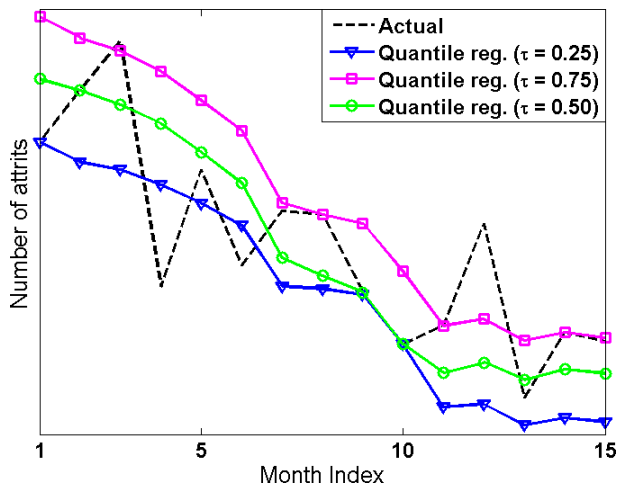


Figure 1: The actual attrition and reconstructed attrition of employees in a large organization obtained using quantile regression at the quantiles $\tau = \{0.25, 0.5, 0.75\}$ [34]. The estimates at $\tau = 0.25$ and $\tau = 0.75$ provides an empirical, non-parametric confidence interval for the median estimate.

frameworks in the literature [1, 15, 16, 37, 43, 29] are customized to the case where ρ_1 is the ℓ_2 loss function. Some applications where dictionaries inferred with the ℓ_1 misfit loss have been effective include robust background modeling [38], emerging topic detection [25], and novel document identification [26].

In this paper, we explore and develop a flexible dictionary learning and sparse coding framework, allowing ρ_1 to be a member of a class of functions rich enough to address real-world challenges. Such a class should include

1. robust penalties, for cases where data may be contaminated by outliers
2. asymmetric penalties, to allow differential treatment of positive and negative elements of the residual vector $r = y - Da$
3. block-assignable penalties, that can act differently on specified subsets of the residual vector

All of these goals can be achieved by considering the general class of piecewise linear quadratic (PLQ) penalties [36, Definition 10.20], which comprise convex penalties whose domain can be represented as the union of finitely many polyhedral sets, relative to which the penalty can be expressed as a general (convex) quadratic. This is a wide class that contains robust penalties such as ℓ_1 , Huber, and Vapnik, asymmetric penalties such as quantile [27, 41], and quantile Huber [5], as well as the classic $\|\cdot\|_2$ penalty, which we refer to as ℓ_2 . Some important PLQ penalties are shown in Figure 2. More details about our proposed framework are available in Section 2. Note that learning a dictionary will also be referred to as learning a sparse model in this paper, and without loss of generality, we will assume that the regularization ρ_2 is the ℓ_1 measure.

1.1 Going beyond ℓ_2 loss functions

From a probabilistic viewpoint, if the elements of the residual follow a Gaussian distribution, it is useful to impose

a squared error or ℓ_2 penalty, which is arguably the most widely used loss function. However, if we expect outliers in the data, a robust loss function should be imposed; and in general, a proper choice of loss function is necessary to estimate the parameters in a noise-robust manner, and in turn lead to improved predictive power for future data. Some straightforward examples are: (a) In econometrics where we model the market value of the company as a linear combination of various accounting numbers [6] and there could be a few bad years for the company due to events like economic depression, (b) In image processing where a few pixels are corrupted due to saturation noise from sensors.

Allowing the loss function ρ_1 to be asymmetric gives us the ability to penalize positive and negative components of the residual $r = y - Da$ differently. Quantile loss is a well-known convex asymmetric function that has been used extensively in regression. It is used to understand and predict the response of a process at various quantiles. For example, the time-varying attrition of a workforce in a company can be posed as a regression problem over incentive variables, and quantile regression allows us to predict the future attrition at various quantiles [34]. For planning purposes, management can use best-case and worst case attrition at high and low quantiles, respectively. The actual attrition, along with median, high (0.75) and low (0.25) quantile estimates over a period of time for a particular company are provided in Figure 1. In addition, predictions at quantiles 0.75 and 0.25 can be used to obtain the interquartile range [45], which is a robust measure of statistical dispersion. This provides us with non-parametric, distribution free confidence limit estimates. Using our proposed approach, dictionaries can be obtained at various quantiles, which can be then used to obtain such estimates for the predictions.

The case for using a block-assignable penalty for ρ_1 may arise when the observation y itself is heterogeneous. ρ_1 then needs to be constructed to penalize some elements of the residual r differently than the others. For example, in image analysis applications, the overall feature y obtained from an image may be a combination of multiple features, each of which require a different loss. There could also be situations where a different noise model applies to each component of y , and hence different loss functions are needed. This is the case in predicting tags for images [], where y is a combination of image level features (where ℓ_2 loss can be used) and user tags (where ℓ_1 loss is required because of the possibility of sparse errors).

1.2 Contributions

In this paper, we propose a dictionary learning framework for the general class of piecewise linear quadratic (PLQ) penalties, and show that the *sparse code update problem* for any formulation with PLQ measurement, regularization, and graph structure penalties can be solved with a recently developed solver [4]. For the generalized PLQ approach to the full dictionary learning problem, we implement a block-coordinate scheme and prove convergence under the assumption that the measurement PLQ penalty is differentiable. Just as in the classic dictionary learning framework, this requires alternating minimization in the sparse codes A and dictionary D . For the latter problem, we use block-coordinate descent to update the columns using an efficient L-BFGS method with Barzilai-Borwein step-length selection. It is important to note that while this method is

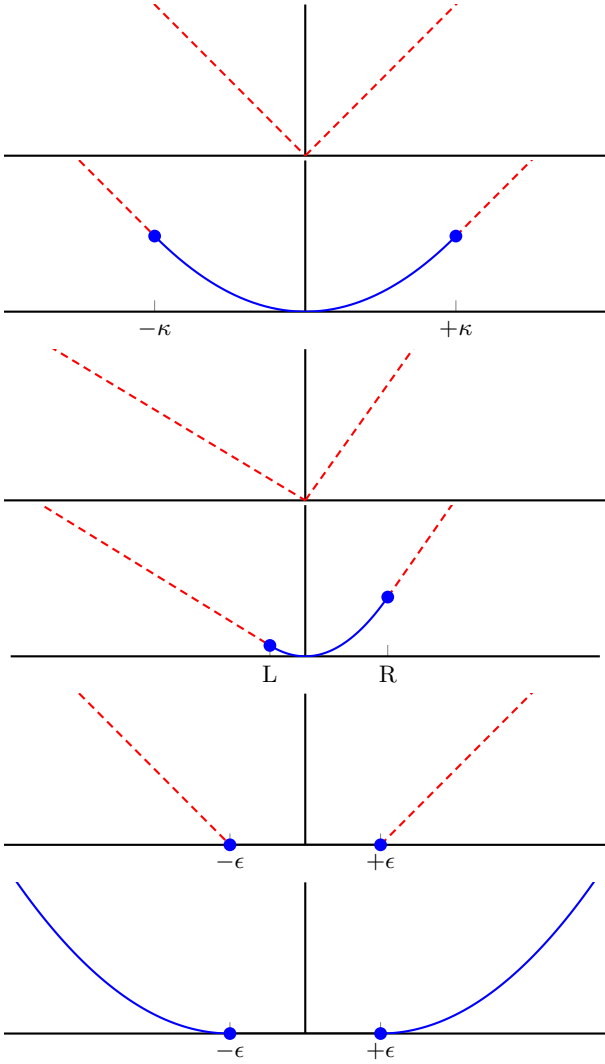


Figure 2: Examples of PLQ penalties for dictionary learning, from top to bottom: ℓ_1 -penalty, huber, quantile loss (0.3), quantile huber (0.3) [5], Vapnik, smooth insensitive loss.

general, when ρ_1 is the quadratic loss, it converges in two steps (which is almost as efficient as a closed-form update rule).

To enable practitioners to develop and test new kinds of PLQ penalties, we extended the interface of [2] to allow (1) different PLQ penalties for different blocks of a residual vector, and (2) automatic Moreau-Yosida smoothing of arbitrary PLQ penalties. The latter feature ensures convergence of block coordinate descent, if applied to any (potentially non-smooth) PLQ formulation. These extensions are communicated as theoretical lemmas related to conjugate representation calculus.

To illustrate the utility of the proposed approach, we apply the algorithm in three different real-world scenarios and provide experimental evaluations. The first scenario concerns the robust modeling of images corrupted by sparse noise. In this case, we train a dictionary for sparse coding of the patches, taking ρ_1 to be the robust Huber penalty. The dictionary is then used to reconstruct the patches, and this

allows much better noise rejection compared to recovery obtained with ρ_1 set as the ℓ_2 penalty. In the second case, we consider the problem of refining human annotated tags in an image data set. Since tags of similar images will be similar, we perform a joint sparse coding of features and tags using the mixed PLQ penalty, with ℓ_2 penalty for features and Huber penalty for tags. The mixed penalty provides more robust estimates compared to using ℓ_2 for both features and tags, at varying levels of impulse noise. In the third application, we evaluate the performance of subspace clustering using ℓ_1 graphs [10] in various data sets at multiple quantiles. The performance of different quantiles around the median can be used to evaluate empirical confidence bounds on the median accuracy.

2. ALGORITHMIC FORMULATION

We begin by formulating a generalized batch dictionary learning problem:

$$\begin{aligned} \min_{A, D} \quad & \rho_1(Y - DA) + \rho_2(A) + \rho_3(A) \\ \text{subject to} \quad & A \in \mathcal{A}, D \in \mathcal{D}. \end{aligned} \quad (2)$$

where $Y = [y_1 y_2 \dots y_T]$ is the observation matrix, $A = [a_1 a_2 \dots a_T]$ is the corresponding sparse code matrix, ρ_1 is the misfit loss function ρ_2 is the sparsity regularization, and ρ_3 encodes other prior information about the codes (for example, graph structure). The constraints $A \in \mathcal{A}$ and $D \in \mathcal{D}$ allow us to encode other prior information about the codes and the dictionary; for example, the columns of the dictionary may be normalized, while all codes may be non-negative.

This problem is nonconvex, and is typically solved in a block-coordinate descent fashion, where the dictionary D and codes A are updated in turn, with the other held fixed. Note that for fixed D , the problem of updating A fully decouples when ρ_1 is the Frobenius norm, and every column of A may be updated in parallel. This generalizes perfectly for any loss function ρ_1 that can be written as a sum of penalties across columns; we assume our loss functions have this property. We refer to this problem as the *code update problem*. For fixed A , the problem of updating D requires consideration of ρ_1 only, together with the constraint $D \in \mathcal{D}$. We refer to this as the *dictionary update problem*.

In this paper, we propose a modeling framework and optimization scheme that is general enough to handle all of these requirements, as well as simple constraints on A and D . Specifically, we allow ρ_1 to come from the class of piecewise linear quadratic (PLQ) penalties, or a mixture of several PLQ penalties. Recently, [4] showed that a broad subclass of these penalties can be given a natural statistical interpretation, and used their conjugate representation to devise a generic interior point method for their solution. This method also efficiently incorporates simple constraints $A \in \mathcal{A}$. We use this method to solve the code-update problem.

In the next section, we review the general class of PLQ penalties, and characterize some properties of this class that make it particularly useful for modeling specific applications. We then specify the representations of penalties we present in our experimental section. We then discuss our method of solving the dictionary update problem, and consider convergence of the entire scheme.

2.1 Piecewise Linear-Quadratic penalties

We briefly review the class of quadratic support (QS) functions, referring the reader to [4] for a full exposition. Every penalty in this class can be written as a convex conjugate to a quadratic function on an arbitrary set:

DEFINITION 2.1. A QS function is any function $\rho(U, M, b, B; \cdot)$ mapping from \mathbb{R}^n to $\mathbb{R} = \mathbb{R} \cup \infty$ having representation

$$\rho(U, M, b, B; y) = \sup_{u \in U} \{ \langle u, b + By \rangle - \frac{1}{2} \langle u, Mu \rangle \}, \quad (3)$$

where $U \subset \mathbb{R}^m$ is a nonempty convex set, $M \in \mathcal{S}_+^n$ the set of real symmetric positive semidefinite matrices, and $b + By$ is an injective affine transformation in y , with $B \in \mathbb{R}^{m \times n}$, so, in particular, $m \leq n$ and $\text{null}(B) = \{0\}$.

If the set U is taken to be polyhedral, i.e. having the representation

$$U := \{u | Cu \leq c\},$$

then the associated QS function becomes piecewise linear quadratic, and is written $\rho(C, c, M, b, B; y)$.

The ability to represent PLQ penalties through the set of structures (A, a, M, b, B) gives rise to a representation calculus, where addition, affine composition, and other manipulations can be done using the underlying structures. We highlight three results that are particularly useful in encoding variants of the code update problem.

LEMMA 2.2 (ADDITION). Let $\rho_1(y)$ and $\rho_2(y)$ be two PLQ penalties specified by C_i, c_i, M_i, b_i, B_i , for $i = 1, 2$. Then the sum $\rho(y) = \rho_1(y) + \rho_2(y)$ is also a PLQ penalty, with

$$C = \begin{bmatrix} C_1 & 0 \\ 0 & C_2 \end{bmatrix}, a = \begin{bmatrix} a_1 \\ a_2 \end{bmatrix}, M = \begin{bmatrix} M_1 & 0 \\ 0 & M_2 \end{bmatrix}, b = \begin{bmatrix} b_1 \\ b_2 \end{bmatrix}, B = \begin{bmatrix} B_1 \\ B_2 \end{bmatrix}.$$

This result is used both to combine measurement and regularization terms into a single representation.

LEMMA 2.3 (AFFINE COMPOSITION). PLQ penalties are closed under affine composition, with

$$\rho(C, c, M, b, B; Dy + d) = \rho(C, c, M, b + Bd, BD; y).$$

This result allows automatic composition of simple building blocks with linear maps.

LEMMA 2.4 (PRODUCT ACTION). A PLQ $\rho(y) = \rho_1(y_1) + \rho_2(y_2)$, where y_1 and y_2 are sub-blocks of the vector y , is easily written in terms of addition and affine composition; namely

$$\rho(y) = \rho_1(M_1 y) + \rho_2(M_2 y),$$

where $M_1 y = y_1$ and $M_2 y = y_2$.

This last result makes it easy to use different penalties on different variable or residual subsets, which is important in some of our applications.

2.2 Optimizing PLQ penalties with polyhedral constraints

Lemmas 2.2, 2.3 and 2.4 show that inference problems involving sums, affine compositions, and coordinate-wise different PLQ penalties can at the end of the day be written down as a minimization problem in the primal variable y

and the conjugate variable u . To this optimization problem we now add polyhedral inequality constraints $Ay \leq a$, obtaining the most general model problem:

$$\min_{y, u} \rho(C, c, M, -b, -B; y) \quad \text{s.t.} \quad Ay \leq a.$$

Note that a simple evaluation of our function at a candidate point y requires partial minimization with respect to the conjugate variable u . It may therefore seem that we have made the problem more complicated; however, keep in mind that (a) if we choose component PLQ penalties from a wide set of common candidates, we have alternative representations for $\rho(y)$ at our disposal, and (b) the conjugate representation has been introduced for the purpose of obtaining a minimum in y .

Through the conjugate representation, we are able to write the Karush-Kuhn-Tucker system of optimality conditions for the entire class of interest. KKT systems are often used to characterize optimality of optimization programs and design algorithms; the advantage of the conjugate representation is that we have a uniform approach to characterizing a wide variety of nonsmooth optimization programs. The details are encoded within the representation, which is formed automatically from individual components using the calculus we have described in the previous section. The KKT system is

$$\begin{aligned} 0 &= B^T u + Aw \\ 0 &= b + By - Mu - Aq \\ 0 &= C^T u + s - c \\ 0 &= A^T y + r - a \\ 0 &= q_i s_i, \quad i = 1, \dots, \ell, \quad q, s \geq 0, \\ 0 &= w_i r_i, \quad i = 1, \dots, \ell, \quad w, r \geq 0, \end{aligned} \quad (4)$$

where s, r are nonnegative slack variables that turn inequality constraints $C^T u \leq c$ and $A^T y \leq a$ into equations, while q, w are the dual variables corresponding to the resulting equality constraints. With (4) in hand, the problem can be solved by relaxing the complementarity slackness conditions (last two equations), and using a damped Newton's method to directly optimize the relaxed system. Full convergence theory for problem without inequality constraints is presented by [4], and [3] shows how constraints can be included.

In the context of dictionary learning, any PLQ penalty can be used for the sparse code update problem. Since we directly solve the KKT system 4 using the method of [4], we always have direct access to an optimality certificate; namely the KKT system itself. However, these guarantees only hold for the (convex) code update problem, and in the next section, we discuss the overall (nonconvex) approach to dictionary learning, and focus on the dictionary update problem.

2.3 Block coordinate descent

We now consider the full nonconvex problem (2). A natural approach is to alternate between updating spare codes A and the dictionary D , which is an instance of block coordinate descent. When the penalties ρ_1, ρ_2 and ρ_3 are smooth, standard convergence results for block coordinate descent can be obtained with e.g. [8, Proposition 2.7.1]). However, in most sparse dictionary learning, ρ_2 is taken to be non-smooth, usually the ℓ_1 norm. In addition, we are interested

in a general theory that applies to the entire PLQ class.

Block-coordinate descent for a class of problems general enough to accommodate our framework is studied in [44]. The main theorem of [44] still depends on a smoothness condition for ρ_1 , and, unfortunately, as [44] points out, this condition is in some sense sharp, because block coordinate descent can fail to converge if only convexity in each block is required.

We now present the main theorem for batch dictionary learning in the PLQ setting.

THEOREM 2.5. *Suppose that ρ_1 in (2) is differentiable, ρ_2 and ρ_3 are convex, and the sets \mathcal{D} and \mathcal{A} are convex. Then block coordinate descent (alternating minimization in A and D) converges to a stationary point of (2).*

PROOF. By assumption, ρ_1 is differentiable on its effective domain; furthermore, the entire objective is convex in A . By [44, Lemma 3.1 and Theorem 4.1(b)], every cluster point of the sequence generated from block-coordinate descent is a stationary point of (2).

From the application perspective, we claim that the requirement that ρ_1 be smooth is not particularly limiting. To understand why, recall that the behavior of a penalty at the origin has the strongest influence on *sparsity* properties; in particular, this is why the ℓ_1 penalty is a key choice for ρ_2 . In contrast, ρ_1 acts on *residuals*; the implication is that choosing ρ_1 that are non smooth (say at the origin) means we will fit a lot of data points *exactly*. While this is potentially useful in some applications, it is not in others; [5] recently demonstrated that a smoothed version of the quantile penalty called the *quantile huber* can outperform the standard quantile penalty in the sparse regression setting.

The next natural question is, suppose we are given a PLQ penalty candidate for ρ_1 which is not smooth; is there a disciplined procedure we can use to smooth it, and remain in the class? Amazingly, it turns out that any PLQ penalty can be easily smoothed using a Moreau envelope, and moreover, application of this technique can be represented using the same calculus we relied on in Lemmas 2.2, 2.3, and 2.4.

Recall that the Moreau envelope of a convex function g can be defined as follows:

$$e_\gamma g(y) = \min_x \frac{1}{2\gamma} \|x - y\|^2 + g(x). \quad (5)$$

From the definition, it is immediately clear that $g_\rho(y)$ is well defined, and is always lower than g , because y is feasible in (5). Finally, it can be easily shown [36] that $g_\rho(y)$ is differentiable, with gradient given by

$$\nabla g_\gamma(y) = \frac{1}{\gamma}(y - \bar{x}),$$

where \bar{x} is the unique minimizer of (5), and γ is a smoothing parameter. The Moreau-Yosida envelope function is also known as the prox operator, and plays a major role in optimization formulations for many signal processing applications, c.f. [11]. As $\gamma \downarrow 0$, the envelope function converges to g in an epigraphical sense.

The salient feature for us is that PLQ penalties are closed under Moreau-Yosida smoothing, and the envelope function is precisely captured by our PLQ representation, as shown in [9, Proposition 4.11], reproduced here for convenience in our notation:

THEOREM 2.6. *Let $\rho(c, C, M, b, B; y)$ be any PLQ penalty. Then the envelope function $e_\gamma \rho(y)$ is also a PLQ penalty*

with representation $\rho(c, C, \bar{M}, b, B; y)$, where

$$\bar{M} = M + \gamma BB^T.$$

This theorem allows us not only to smooth any member of the PLQ class by a particular amount γ , but to obtain the representation of the resulting function in closed form. The power of this idea is shown by the following corollary:

COROLLARY 2.7. *The envelope of the 1-norm, $e_\gamma \ell_1$, is the huber function with threshold μ .*

In particular, the ‘smoothing’ of the ℓ_1 to get the huber is an idea that easily generalizes to the entire class, and is captured by the conjugate representation calculus.

2.4 Dictionary update problem

In the previous section, we simply claimed to solve the dictionary update problem

$$\min_D \rho_1(Y - DA)$$

for a fixed set of sparse codes A . We now explain how we solve this problem, and prove the convergence of our scheme. In the least squares case, it is straightforward to implement a block-coordinate optimization scheme on the columns of D , obtaining closed-form updates as we loop over the columns.

In the general case, up pose that we wish to update the j -th column of D_j . Letting a_j denote the j th row of A , d_j denote the j th column of D , and $D_{/j}$ to denote the dictionary with the j th column deleted, we it is easy to see that

$$DA - D_{/j}A = d_j a_j^T.$$

For penalties ρ which decompose over the columns of the residual $Y - DA$, the optimization formulation to d_j is given by

$$\bar{d}_j = \min_d \rho_1(Y_j - d a_j^T). \quad (6)$$

with $Y_j = Y - D_{/j}A$. For the least-squares case, this update problem has a closed form solution; and in the general case, the structure of the problem is very simple: the k th entry of $d_{j,k}$ is determined by solving a scalar optimization problem

$$d_{j,k} = \operatorname{argmin}_{d_k} \rho_1(Y_j(k, \cdot) - d_k a_j).$$

Since this is a 1-dimensional optimization problem, the Barzilai-Borwein [7] line search method is equivalent to Newton’s method in the quadratic case (after 2 steps). Motivated by this, we use L-BFGS with Barzilai-Borwein line search to solve (6). For quadratic ρ_2 , this method converges in two iterations per column, as expected, and for general smooth ρ_2 , such as the Huber, it is also rapidly convergent. Since Theorem 2.5 requires ρ_1 to be smooth, block-column coordinate descent converges by [8, Proposition 2.7.1].

The empirical convergence of the overall block-coordinate descent scheme for the proposed batch dictionary learning problem is shown in Figure 3, for a real-world data set. Clearly, the ℓ_2 loss has the fastest convergence when compared to Huber and quantile Huber ($\tau = 0.25$), since it is the most well-behaved among all the three.

3. EXPERIMENTS

Added flexibility in the choice of loss functions can make sparse models more effective in several applications. In this

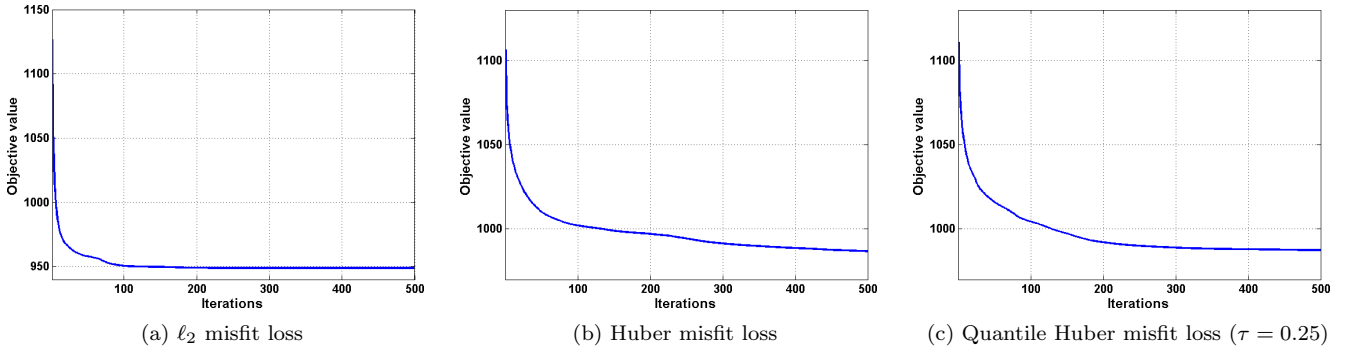


Figure 3: Empirical convergence of the block-coordinate descent scheme for the proposed batch dictionary learning problem for various misfit losses.

section, we present three different case studies that demonstrate the importance of adopting a general framework. Though more sophisticated applications of sparse models can be considered, our emphasis is on illustrating the flexibility and robustness of the proposed framework in comparison to conventional sparse modeling approaches. We conclude this section by briefly discussing possible extensions to this work, and relevant applications that can benefit from the general framework developed in this paper.

3.1 Scenario 1: Robust Image Modeling

The statistics of natural images motivates the use of sparse models to describe them [17], and makes it possible to recover them from different forms of corruption. For simplicity, we consider corruption by an additive noise, which can happen during sensing or transmission. In such scenarios, a generalizable model should ignore the underlying noise, and describe only the relevant patterns in the image. Such a robust model can then be used to denoise the image or improve its quality. When the noise is Gaussian, the traditional sparse modeling framework, which uses an ℓ_2 loss function, can be very effective in discovering patterns that are masked by noise. However, when the noise model is non-Gaussian, the sparse model learned using this procedure will no longer be robust.

Consider the case where an image is corrupted by salt-and-pepper noise, which manifests as randomly occurring white and black pixels in the image. A typical noise reduction strategy for this kind of noise is to apply median filtering. Therefore, we propose to use the Huber penalty as the loss function, since it can learn median patterns in the dictionary, thereby resulting in a robust model. However, using an ℓ_2 loss function will infer noisy patterns, since it tends to spread the noise through the pattern, and hence the denoising performance using the dictionary will also be poor.

Given an image I , we extract non-overlapping patches of size 8×8 , vectorize these patches, and stack them to form a matrix denoted X . Adding salt-and-pepper noise at a specific percentage is equivalent to randomly replacing the percentage of pixels with a black or a white pixel. In this experiment, we vary the noise level from 1% to 15%. We learn dictionaries using different penalties, and compare the reconstruction obtained using the learned sparse model with the original clean image. When the model is robust, we expect that the impulse noise will not be a part of the dic-

tionary elements, and hence the reconstruction will be of high quality. Note that we do not perform any explicit denoising, and only evaluate the quality of the reconstruction from the model. We measure the peak-signal-to-noise ratio (PSNR) for the noisy image, and the images recovered using sparse models learned with ℓ_2 and Huber penalties. Figure 4 shows the results obtained for increasing levels of impulse noise, and the robustness of the Huber penalty is clearly evident from the higher PSNR values, as well as from the improved visual quality.

3.2 Scenario 2: Refining Tags for Image Retrieval

Textual descriptors, or *tags*, are useful meta-data for images in retrieval applications. In large scale retrieval systems, it is typical to present a textual query to retrieve semantically relevant images. Since a single semantic concept can manifest into a wide range of visual representations, it is often difficult to mine a database using just visual features or tags. Furthermore, human annotation can be very subjective and error-prone. The goal of automatic image annotation is to predict new tags, and possibly refine existing noisy tags, based on information from visually similar images. In this experiment, we will consider the problem of refining the noisy tags of a novel image using a set of training images. For each image, a tag vector is typically a binary vector that indicates the relevance of each semantic topic from a pre-defined vocabulary. Due to human errors, or the limitation of prediction systems, some unrelated concepts could be included in the image description, or important topics could be left out. Sparse or low-rank models learned using both visual features and the noisy tags can be very effective in refining the semantic descriptors [46, 48, 49].

Given a set of training images, we use the *Gist* features [32] to describe the visual content. The set of visual features are stored in the matrix X , and their corresponding textual descriptors are stored in the matrix B . Given a novel image feature (*Gist*) y , and its noisy tag vector h , our goal is to obtain a refined estimate \hat{h} . We propose to exploit the correlations between the features and tags, using sparse coding, to perform tag refinement. Using the set of training examples, we construct the dictionary $D = [X^T \gamma B^T]^T$, where γ is the scaling factor used to balance the total energy of features and tags. Similarly, the test sample is described as $z = [y^T \gamma h^T]^T$. By assuming that the features and tags are clustered along subspaces, this structure can be discovered

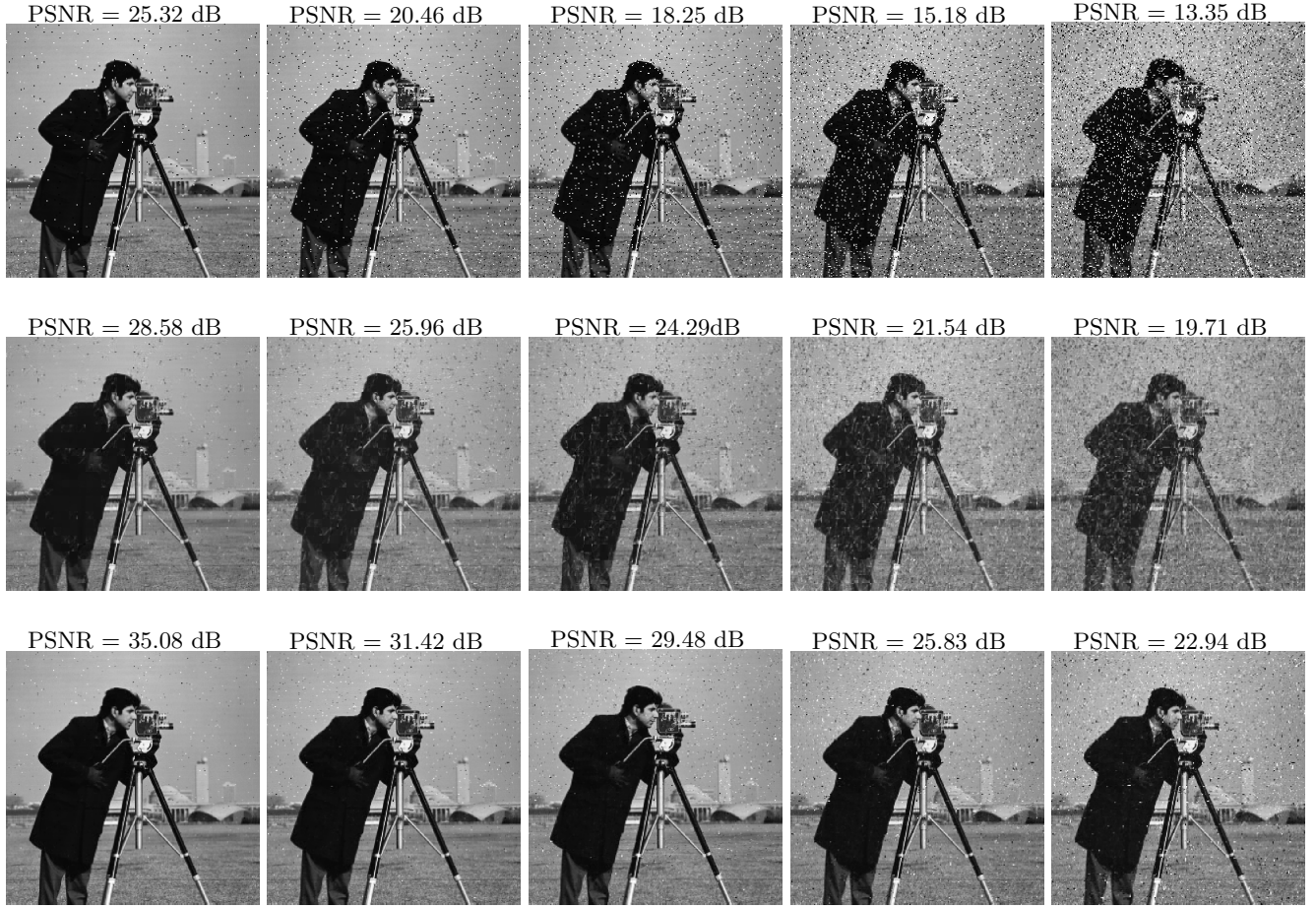


Figure 4: Robust Image Modeling - Row 1 shows images corrupted by increasing levels of salt and pepper noise. Rows 2 and 3 show the images recovered using sparse models learned with the ℓ_2 and Huber penalties, respectively. In each case, the corresponding PSNR (dB) value is also reported. The robustness achieved by considering a more appropriate loss function when the corruption is non-Gaussian is clearly evident.

using sparse coding on examples:

$$\min_a \|z - Da\|_2^2 + \lambda \|a\|_1. \quad (7)$$

The refined tag vector can then be estimated as $\bar{h} = Ba$. The formulation in (7) assumes that both features and semantic descriptors can be recovered using the same set of sparse coefficients. However, the ℓ_2 penalty is not robust, and thus unsuitable for measuring the misfit in the reconstruction of tag vectors. To improve the recovery, we use different penalties for modeling visual features and tag vectors:

$$\min_a \rho_1^{(1)}(y - Xa) + \rho_1^{(2)}(h - Ba) + \lambda \|a\|_1, \quad (8)$$

where $\rho_1^{(1)}$ is the ℓ_2 penalty, and $\rho_1^{(2)}$ is the Huber penalty. As discussed in Section 2, the proposed framework can use the resultant mixed penalty to obtain sparse codes.

For our experiment, we used the Corel-5K data set [13], which is a very commonly used comparative data set for image annotation. There are 5,000 images in total, and each image is annotated with 1 to 5 keywords. We used 4,500 images as training data, and evaluated the performance using the rest. The total number of keywords in the vocabulary is 260. We varied the level of noise in the test tags, by

randomly flipping $\{1\%, 3\%, 5\%, 10\%, 15\%, 20\%, 25\%\}$ of the entries in each binary tag vector. We estimated the refined tags using the both schemes described earlier, and computed the average noise (%) in the refined tag vectors. Figure 5(a) plots the performance obtained using the ℓ_2 penalty for the entire residual, and the mixed ℓ_2 -Huber penalty. As can be seen, the robust variant using the mixed penalty provides improved recovery at all noise levels. Furthermore, we corrupted the tag vectors of the training data also with different levels of noise and studied the performance deterioration (Figures 5(b) and (c)). We found that using mixed penalties provided superior performance in all cases.

3.3 Scenario 3: Computing Empirical Confidences for Subspace Clustering

Assuming that the data samples lie in a union of subspaces allows us to perform unsupervised clustering using sparse coefficients [35, 10]. By constructing a suitable graph to describe the relationship between data samples, we can analyze the eigen spectrum of the graph Laplacian to determine the underlying clusters. In particular, we can build an ℓ_1 graph [10] for unlabeled data by solving for sparse codes using the data samples as the dictionary, with the constraint

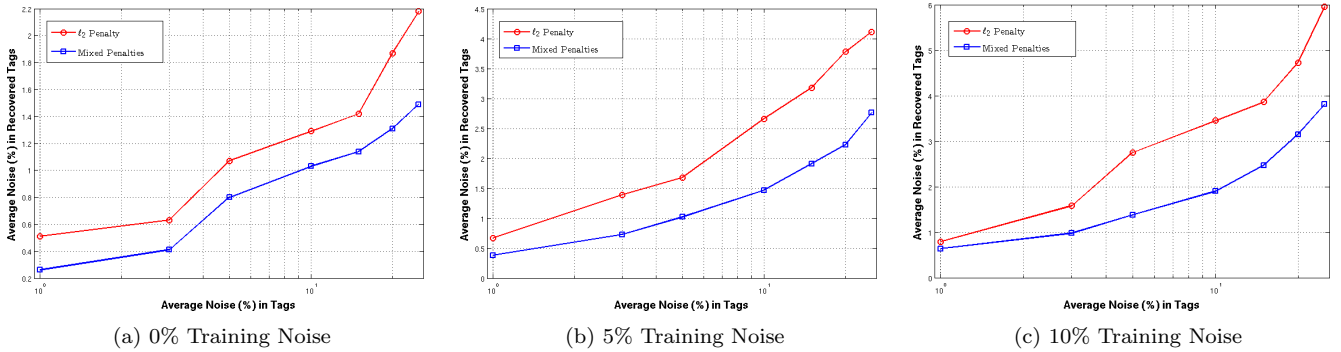


Figure 5: Tag Refinement using ℓ_2 (red) and mixed ℓ_2 –Huber penalties (blue). Using appropriate robust penalties for the tags result in improved recovery performance at all noise levels for training data.

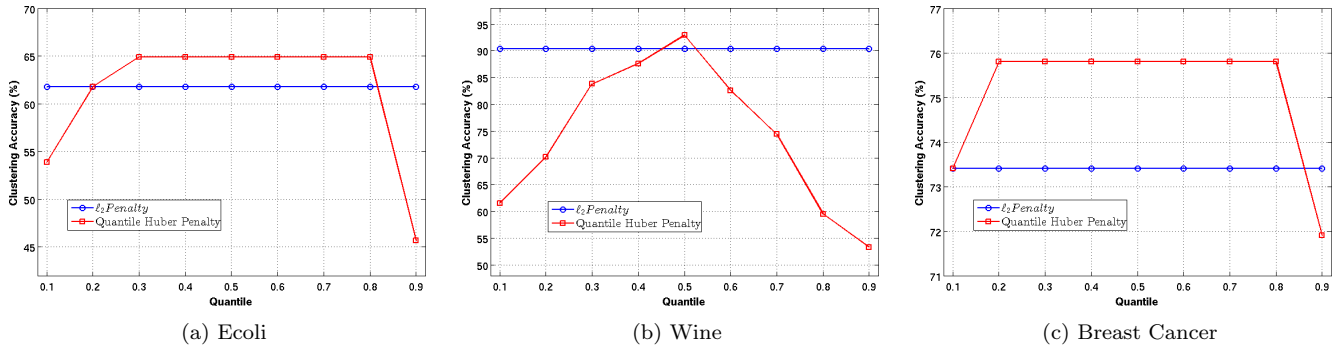


Figure 6: Evaluating the performance of subspace clustering using ℓ_1 -graphs at various quantiles for different datasets. The quantile-specific performances can be used to obtain empirical confidence limits on the median performance. The clustering performance with ℓ_2 loss (blue) is also provided for comparison with quantile huber results (red).

that a sample cannot contribute to its own representation. This procedure provides a non-local graph, as opposed to other locality-based graph construction strategies such as k-nearest neighbors. The coefficient matrix $A \in \mathbb{R}^{T \times T}$ from sparse coding is used to construct the graph Laplacian as $(\mathbb{I} - A)^T(\mathbb{I} - A)$, where \mathbb{I} is the identity matrix. This model is very effective in several scenarios, but using the ℓ_2 loss function to obtain the sparse code matrix A makes it highly non-resilient to outliers. Furthermore, it is not possible to estimate confidence measures on the clustering performance using the ℓ_2 loss function. We propose to employ the quantile Huber penalty to overcome these shortcomings, and generate reliable clusterings. Using the τ^{th} quantile to measure the misfit is equivalent to allowing a fraction τ of the entries in the residuals to be positive. Unless there is a complete model misfit, the penalty will deteriorate gradually as we consider quantiles away from the median. It was observed in the previous case studies that using the Huber (0.5 quantile) penalty makes sparse models robust to the outliers. In order to derive empirical confidence intervals and understand the reliability of clustering, we generate multiple ℓ_1 graph based clusterings at different quantiles, and study their corresponding clustering performances.

In this experiment, we consider three datasets from the UCI repository: (i) Ecoli, (ii) Wine, and (iii) Breast cancer datasets. In each case, we build ℓ_1 graphs at multiple quantiles between 0.1 and 0.9 and evaluate the clustering performance as the % Accuracy. Figure 6 illustrates the clustering performances for the three datasets. In each case, we report

the performance obtained by using the ℓ_2 penalty for comparison. In addition to improving the clustering accuracy, using a more flexible loss function enables us to understand the reliability of the clustering results. For example, in case of the *Wine* dataset (Figure 6(b)), though the median performance is high, the performance drops significantly as we move away from the median quantile. This shows that the clustering is very sensitive to the outliers, and small perturbations to the dataset might result in a sub-optimal performance. This behavior can be attributed to the limited availability of samples, or non-suitability of the chosen generative model. Though the union of subspace assumption seems valid for this data, lack of high confidence suggests that we choose different model assumptions for clustering.

4. CONCLUSIONS

Analyzing complex, high-dimensional data requires the design of interpretable, robust and scalable models. The proposed general framework has the inherent advantage of allowing one to compute sparse codes and optimize dictionaries using a broad class of misfit losses. However, the challenges in scaling this framework to large scale settings, and the inability to perform warm-starts with interior point methods make the design of online learning algorithms difficult. Using proximal methods, in lieu of interior point solvers, is a possible approach to overcome these shortcomings. In addition to enabling fast sparse code computation using warm-starts and online dictionary inference, this will

allow us to go beyond sparsity regularization, and incorporate other penalties such as the nuclear norm (low-rank) regularization. Another important extension is to evaluate the proposed framework using graph penalties on the sparse codes for batch learning. It will be useful to incorporate similar penalties in online learning as well. We believe that such a general and scalable framework can expand the applicability of sparse models in data analytics. Some important application areas that can benefit by this development include matrix completion for recommender systems, topic modeling in text analytics, analysis of interactions in large networks, semantic content analysis in images/videos and data visualization.

5. REFERENCES

- [1] M. Aharon, M. Elad, and A. Bruckstein. K-SVD: An algorithm for designing overcomplete dictionaries for sparse representation. *IEEE TSP*, 54(11):4311–4322, November 2006.
- [2] A. Y. Aravkin. ipsolve: An interior point solver for plq penalties, 2012. <https://github.com/saravkin/IPsolve>.
- [3] A. Y. Aravkin, J. V. Burke, and G. Pillonetto. Linear system identification using stable spline kernels and plq penalties. *To appear in IEEE Conf. Decision and Control (CDC), March 2013*, 2013.
- [4] A. Y. Aravkin, J. V. Burke, and G. Pillonetto. Sparse/robust estimation and kalman smoothing with nonsmooth log-concave densities: Modeling, computation, and theory. *Journal of Machine Learning Research*, 14:2689–2728, 2013.
- [5] A. Y. Aravkin, A. Kambadur, A. C. Lozano, and R. Luss. Sparse Quantile Huber Regression for Efficient and Robust Estimation. Technical report, IBM T.J. Watson Research Center, <http://arxiv.org/abs/1402.4624>, 2014.
- [6] M. E. Barth, W. H. Beaver, and W. R. Landsman. Relative valuation roles of equity book value and net income as a function of financial health. *Journal of Accounting and Economics*, 25(1):1–34, 1998.
- [7] J. Barzilai and J. M. Borwein. Two-point step size gradient methods. *IMA Journal of Numerical Analysis*, 8(1):141–148, 1988.
- [8] D. P. Bertsekas. *Nonlinear Programming*. Athena Scientific, second edition, 1999.
- [9] J. V. Burke and T. Hoheisel. Epi-convergent smoothing with applications to convex composite functions. *SIAM Journal on Optimization*, 23(3):1457–1479, 2013.
- [10] B. Cheng, J. Yang, S. Yan, Y. Fu, and T. S. Huang. Learning with l1-graph for image analysis. *IEEE transactions on image processing*, 19(4):858–66, apr 2010.
- [11] P. L. Combettes and J. C. Pesquet. Proximal splitting methods in signal processing. In *Fixed-Point Algorithms for Inverse Problems in Science and Engineering*, pages 185–212. Springer, 2011.
- [12] D. L. Donoho. Compressed sensing. *Information Theory, IEEE Transactions on*, 52(4):1289–1306, 2006.
- [13] P. Duygulu, K. Barnard, J. F. de Freitas, and D. A. Forsyth. Object recognition as machine translation: Learning a lexicon for a fixed image vocabulary. In *Computer Vision—ECCV 2002*, pages 97–112. Springer, 2002.
- [14] M. Elad. *Sparse and Redundant Representations: From Theory to Applications in Signal and Image Processing*. Springer, 2010.
- [15] K. Engan, S. O. Aase, and J. H. Husoy. Method of optimal directions for frame design. In *IEEE ICASSP*, 1999.
- [16] K. Engan, B. Rao, and K. Kreutz-Delgado. Frame design using FOCUSS with method of optimal directions (MoD). In *Norwegian Signal Processing Symposium*, 1999.
- [17] D. Field. Relations between the statistics of natural images and the response properties of cortical cells. *Journal of the Optical Society of America*, 4:2379–2394, 1987.
- [18] J. F. Gemmeke, T. Virtanen, and A. Hurmalainen. Exemplar-based sparse representations for noise robust automatic speech recognition. *Audio, Speech, and Language Processing, IEEE Transactions on*, 19(7):2067–2080, 2011.
- [19] D. Giacobello, M. G. Christensen, M. N. Murthi, S. H. Jensen, and M. Moonen. Sparse linear prediction and its applications to speech processing. *Audio, Speech, and Language Processing, IEEE Transactions on*, 20(5):1644–1657, 2012.
- [20] R. Gribonval, S. Lesage, et al. A survey of sparse component analysis for blind source separation: principles, perspectives, and new challenges. In *ESANN’06 proceedings-14th European Symposium on Artificial Neural Networks*, pages 323–330, 2006.
- [21] T. Hastie, R. Tibshirani, J. Friedman, T. Hastie, J. Friedman, and R. Tibshirani. *The elements of statistical learning*. Springer, 2009.
- [22] K. Huang and S. Aviyente. Sparse representation for signal classification. In *NIPS*, 2006.
- [23] J. Wright *et.al.* Robust face recognition via sparse representation. *IEEE TPAMI*, 31(2):210–227, 2001.
- [24] J. Yang *et.al.* Linear spatial pyramid matching using sparse coding for image classification. In *IEEE CVPR*, 2009.
- [25] S. P. Kasiviswanathan, P. Melville, A. Banerjee, and V. Sindhwani. Emerging topic detection using dictionary learning. In *Proceedings of the 20th ACM international conference on Information and knowledge management*, pages 745–754. ACM, 2011.
- [26] S. P. Kasiviswanathan, H. Wang, A. Banerjee, and P. Melville. Online l1-dictionary learning with application to novel document detection. In *NIPS*, pages 2267–2275, 2012.
- [27] R. Koenker and G. Bassett Jr. Regression quantiles. *Econometrica: journal of the Econometric Society*, pages 33–50, 1978.
- [28] Y. Li, A. Cichocki, and S.-i. Amari. Analysis of sparse representation and blind source separation. *Neural computation*, 16(6):1193–1234, 2004.
- [29] J. Mairal, F. Bach, J. Ponce, and G. Sapiro. Online learning for matrix factorization and sparse coding. *The Journal of Machine Learning Research*, 11:19–60, 2010.
- [30] J. Mairal, F. Bach, J. Ponce, G. Sapiro, and

- A. Zisserman. Supervised dictionary learning. *NIPS*, 2008.
- [31] A. Maurer, M. Pontil, and B. Romera-Paredes. Sparse coding for multitask and transfer learning. *arXiv preprint arXiv:1209.0738*, 2012.
- [32] A. Oliva and A. Torralba. Building the gist of a scene: The role of global image features in recognition. *Progress in brain research*, 155:23–36, 2006.
- [33] R. Raina, A. Battle, H. Lee, B. Packer, and A. Y. Ng. Self-taught learning: transfer learning from unlabeled data. In *Proceedings of the 24th international conference on Machine learning*, pages 759–766. ACM, 2007.
- [34] K. N. Ramamurthy, K. R. Varshney, and M. Singh. Quantile regression for workforce analytics. In *Proc. IEEE GlobalSIP*, 2013.
- [35] I. Ramirez, P. Sprechmann, and G. Sapiro. Classification and clustering via dictionary learning with structured incoherence and shared features. In *IEEE CVPR*, pages 3501–3508, Jun. 2010.
- [36] R. Rockafellar and R. Wets. *Variational Analysis*, volume 317. Springer, 1998.
- [37] R. Rubinstein, A. M. Bruckstein, and M. Elad. Dictionaries for sparse representation modeling. *Proceedings of the IEEE*, 98(6):1045–1057, 2010.
- [38] R. Sivalingam, A. D’Souza, M. Bazakos, R. Miezianko, V. Morellas, and N. Papanikolopoulos. Dictionary learning for robust background modeling. In *Robotics and automation (ICRA), 2011 IEEE International Conference on*, pages 4234–4239. IEEE, 2011.
- [39] G. S. Sivaram, S. K. Nemala, M. Elhilali, T. D. Tran, and H. Hermansky. Sparse coding for speech recognition. In *Acoustics Speech and Signal Processing (ICASSP), 2010 IEEE International Conference on*, pages 4346–4349. IEEE, 2010.
- [40] J.-L. Starck, F. Murtagh, and J. M. Fadili. *Sparse image and signal processing: wavelets, curvelets, morphological diversity*. Cambridge University Press, 2010.
- [41] I. Takeuchi, Q. V. Le, T. D. Sears, and A. J. Smola. Nonparametric quantile estimation. *The Journal of Machine Learning Research*, 7:1231–1264, 2006.
- [42] C. Taswell. The what, how, and why of wavelet shrinkage denoising. *Computing in science & engineering*, 2(3):12–19, 2000.
- [43] I. Todic and P. Frossard. Dictionary learning. *Signal Processing Magazine, IEEE*, 28(2):27–38, 2011.
- [44] P. Tseng. Convergence of a block coordinate descent method for nondifferentiable minimization. *Journal of Optimization Theory and Applications*, 109(3):475–494, 2001.
- [45] G. Upton and I. Cook. *Understanding statistics*. Oxford University Press, 1996.
- [46] C. Wang, S. Yan, L. Zhang, and H.-J. Zhang. Multi-label sparse coding for automatic image annotation. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 1643–1650. IEEE, 2009.
- [47] S. Yan and H. Wang. Semi-supervised learning by sparse representation. In *SDM*, pages 792–801. SIAM, 2009.
- [48] S. Zhang, J. Huang, Y. Huang, Y. Yu, H. Li, and D. N. Metaxas. Automatic image annotation using group sparsity. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 3312–3319. IEEE, 2010.
- [49] G. Zhu, S. Yan, and Y. Ma. Image tag refinement towards low-rank, content-tag prior and error sparsity. In *Proceedings of the international conference on Multimedia*, pages 461–470. ACM, 2010.
- [50] M. Zibulevsky and B. A. Pearlmutter. Blind source separation by sparse decomposition in a signal dictionary. *Neural computation*, 13(4):863–882, 2001.